記事連番0032

社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.
NC98-129 (1999-03)

マルチステップ状態予測を用いた強化学習 —注視点移動を考慮した自動車運転モデル—

小池 康晴† 銅谷 賢治‡

†東京工業大学
‡科学技術振興事業団

†〒 226-8503 横浜市緑区長津田町 4259
‡〒 619-0288 京都府相楽郡精華町光台 2-2

E-mail: †koike@pi.titech.ac.jp ‡doya@erato.atr.co.jp

あらまし 本報告では，制御対象の内部モデルによる多ステップの状態予測を用いた強化学習方式を提案する．これを，自動車の運転に応用し，さまざまな曲率を含む道路を走行し，道路から外れたかどうかだけを評価することにより，学習を行った結果について述べる．車両の動力学モデルを用いて，複数の時間ステップで予測された状態をもとに，報酬の予測と制御出力の決定を行なう．シミュレーションの結果，未学習の曲率の道路形状でも，速度においても走行することができることを確認した．この制御方式では，制御の方針や環境によって用いられる外部情報が変化し，運動制御に必要な情報を得るために視線を変化させるモデルと考えることができる．

キーワード 強化学習，フォワードモデル，視線，自動運転

Multiple state estimation reinforcement learning for driving model

Yasuharu KOIKE† Kenji DOYA‡

†Tokyo Institute of Technology
‡Japan Science and Technology Corporation

†4259 Nagatsuda-cho, Midori-ku, Yokohama, Kanagawa 226-8503, Japan
‡2-2 Hikari-dai Seika-cho Soraku-gun Kyoto 619-02, Japan

E-mail: †koike@pi.titech.ac.jp ‡doya@erato.atr.co.jp

Abstract In this report, a multiple state estimation method for reinforcement learning model is proposed. The steering maneuver of a vehicle is learned from a reward. The reward was evaluated which the vehicle is on the road or not. The reward and control signal were calculated by multiple state using a vehicle dynamics model. From the simulation result, this model can drive on unknown road configuration or velocity condition. This model also explain a gaze control by changing information using a control policy or an environment.

key words reinforcement learning, forward model, gaze, stering contorl

1 はじめに

本報告では，自動車の運転操作に関して，さまざまな曲率を含む道路を走行し，道路から外れたかどうかだけを評価することにより，強化学習の枠組を用いて学習した結果について述べる．

運転をしている時に，このまま進むとカーブを曲がり切れないとか，前の車にぶつかってしまうと言った予測を用いていると感じる場面がある．このような予測は，制御対象の順モデルが必要となるが，人間の小脳には内部モデルが学習により獲得され，そのモデルを用いて制御を行なっていると考えられている [1]．さらに，複数の内部モデルを用意し，内部モデルの予測誤差により複数の内部モデルを組み合わせて制御を行なうモデル [2] が提案されている．強化学習の枠組においても，非線形を持つ環境下において，制御則の学習と，それらの切替を組み合わせることで倒立振子制御が行なわれている [3]．

自動車の運転の場合，道路が曲がっているため，この変化に追従する必要がある．滑べりやすい路面，車の大きさや重さなどの物理特性の違い，前進，後退による挙動の違い，速度の違いなどにより車のモデルが複数必要となると考えられる．視覚情報の遅れを補うためや，道路変化を見究めるため，複数の場所での情報を用いて制御を行なっていると考えられる．そのため，一つの内部モデルだけを用いるが，そのモデルにより複数の場所で予測を行ない，それらを組み合わせることで制御を行なう枠組を提案する．

ひとつの場所だけで予測を行なうモデル [4] では，設計者が何秒先の情報を用いるかを試行錯誤で決定していた．また，走行するコースの形状の違いも考慮して，設定時間も決定していた．複数の場所の情報を用いることで，思考錯誤で決めていた時間を評価関数の値により決定することができる．運転中の視線とステアリング操作を同時に計測したり [5]，視野を制限して運転させた時に運転操作がどのように変化するかを調べる研究 [6] が行なわれ，視線とステアリング操作が遅れを伴ってほぼ比例していることや，複数の場所からの情報により操作していることがわかってきた．さらに，人間の視覚的注意の観点から注視点と有効視野の関係が議論され注意の遠近移動が起こることなどが報告されている [7]．これまで，画像の特徴により視線を計算するモデルがほとんどであった [8] が，本報告では，運動制御に必要な情報を得るために視線を変化させるモデルになっており，制御の方針や環境によっても変化する特徴を持つ．

これにより，道路上の障害物を避けたり，追い越したりすることができることも示す．

2 内部モデル

脳の中に内部モデルが存在しているのではないかという実験的な証拠が示され始めており [1, 9]，人間が内部モデルを様々に利用している可能性について考えられている [10]．例えば，外部環境の変化を感覚器で感じとっているが，体も外部環境も動いている場合，自分の動きによる感覚器の変化を内部モデルを用いて予測することで，外部環境の変化だけを取り出すことができる．

また，現在の状態と，運動指令から次の状態を推定する状態予測器としても使用することができる．制御の世界で"observer" と呼ばれているものに対応する．さらに，フィードバックに遅れが存在すると制御系は不安定になるが，追従課題を行なった時の人間の視覚の遅れは，$200-300ms$ となることが知られている．このように大きな遅れがある時に，内部モデルを用いて状態を推定し，推定されたフィードバック信号を用いることで，遅れを補償できる． これは，スミス予測器として知られている．

本報告では，"actor-critic" アーキテクチャーを用いたTD 学習を用いるが，内部モデルを使って，将来の状態を予測することで，

1. 制御回路 (actor) への入力として使う
2. 状態の評価回路 (critic) への入力として使う
3. 複数ステップ

などが可能となる．これにより，より良い Value Function の生成や行動選択が行なえるかについて検討することにした．

2.1 連続版の TD 誤差

システムの状態方程式は，以下のように与えられる．

$$\frac{d\mathbf{x}(t)}{dt} = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{1}$$

ここで，$\mathbf{x} \in X \subset \mathbf{R}^n$ は状態，$\mathbf{u} \in U \subset \mathbf{R}^m$ は，制御入力である．

報酬は，状態と制御入力の関数として与えられる．

$$r(t) = r(\mathbf{x}(t), \mathbf{u}(t)). \tag{2}$$

ある制御即が与えられると，

$$\mathbf{u}(t) = \mu(\mathbf{x}(t)), \tag{3}$$

状態 $\mathbf{x}(t)$ の "value function" は，

$$V^{\mu}(\mathbf{x}(t)) = \int_t^{\infty} \frac{1}{\tau} e^{-\frac{s-t}{\tau}} r(\mathbf{x}(s), \mathbf{u}(s)) ds, \tag{4}$$

ここで，$\mathbf{x}(s)$ と $\mathbf{u}(s)$ $(t < s < \infty)$ は，システムのダイナミクス (1) と制御則 (3) に従う．目標は，どん

な状態 $\mathbf{x} \in X$ でも $V^{\mu}(\mathbf{x})$ を最大にする最適な制御則 μ^* を見つけることである．τ は，discount factor に関係する時定数である（$\lambda = 1 - \frac{\Delta t}{\tau}$）．

式 (4) の t での微分は，

$$\tau \frac{d}{dt} V^{\mu}(\mathbf{x}(t)) = V^{\mu}(\mathbf{x}(t)) - r(t). \quad (5)$$

$P(t)$ を value function $V^{\mu}(\mathbf{x}(t))$ の予測とする．次のエラーを最小にすることで，P_t が value function に等しくなる．

$$\hat{r}(t) = r(t) - P(\mathbf{x}(t)) + \tau \frac{dP(\mathbf{x}(t))}{dt}, \quad (6)$$

この値を連続版での TD error とする．dP/dt は，

$$\hat{r}(t) = r(t) - P(\mathbf{x}(t)) + \tau \frac{P(\mathbf{x}(t)) - P(\mathbf{x}(t-\tau_c))}{\tau_c} \equiv 0,$$

$$P(\mathbf{x}(t-\tau_c)) \equiv P(\mathbf{x}(t)) + \frac{\tau_c}{\tau}(r(t) - P(\mathbf{x}(t))). \quad (7)$$

ここで，τ_c はいかに過去の報酬まで計算するかの時定数である．これは，制御の時間区切り Δt とは，独立して設定できる．

2.2 内部モデルの予測による Actor-Critic 構造

本報告では，"actor-critic" アーキテクチャーを用いて，時間や空間の表現は，連続系とした連続時間の TD 学習 [11] を用いてシミュレーションを行なった．

内部モデルを用いることで，現在の状態 $x(t)$ をもとに，現在の制御入力 $u(t)$ をとり続けた場合の将来の状態 $\bar{x}(t+nT)(n = 1, 2, \cdots)$ を予測することができる．これらを actor,critic の入力として用いることにより，現在の状態を用いるよりも，より予測的で効率の良い制御を行なうことができる可能性がある．しかし，遠い将来の状態予測には不確実性が伴い，どれだけ先の時刻の予測状態を用いるべきかは，その時々の環境の状態に依存する．

そこで，どの時点での状態予測値を用いるべきかを決めるための基準として，予測される累積報酬

$$R^{(0)} = V(x(t)) \quad (8)$$

$$R^{(1)} = \bar{r}(t+T) + \lambda V(\bar{x}(t+T)) \quad (9)$$

$$R^{(2)} = \bar{r}(t+T) + \lambda \bar{r}(t+2T) + \lambda^2 V(\bar{x}(t+2T)) \quad (10)$$

$$\cdots$$

を考える．ここで，$\bar{x}(t+T)$ は，時刻 t から時刻 $t+T$ までの予測された状態に対する報酬の積分値，λ は discount factor である．

すると，n-step の報酬の期待値は，真の報酬の関数 $V^{\pi}(s)$ の近似としての現在の報酬の関数 $V(s)$ よりも良くなる [12]．

$$\max_s \left| E_{\pi}\{R^{(n)} | s_t = s\} - V^{\pi}(s) \right| \leq \lambda^n \max_s |V(s) - V^{\pi}(s)| \quad (11)$$

しかし，これは，予測が確かであるという条件付である．一般に将来にわたって正しく予測できなかったり，環境が変化することで予測が正しくなくなる場合もある．

図 1 に内部モデルによる予測を用いた Actor-Critic 構造の強化学習の枠組を示す．内部モデルを用いて予測した場所で入力情報となる，道の左右の端からの距離や進行方向と道路のなす角やその角速度を，一つの Actor と一つの Critic に入力し，それぞれの場所での Actor と Critic の値を計算する．

つぎに，式 (12) の Softmax により複数の位置で，どこの状態が良いのかを示す分配信号 α を計算する．

$$\alpha_i = \frac{e^{R^i}}{\sum_{j=0}^{n-1} e^{R^j}} \quad (12)$$

この分配信号を用いて，制御出力 u を計算する．

$$u = \sum_{i=0}^{n-1} \alpha_i u_i \quad (13)$$

これにより，将来にわたって報酬が良い場合は，遠くの情報も使って制御を行い，そうでない場合は，Value Function の値にしたがって，近くの情報を主に用いて制御を行う．Actor-Critic の学習は，現時点での報酬をもとに TD 誤差を計算し，この値に基づき重みを更新することにより行なう．

3 シミュレーション

3.1 神経回路モデルによる実装

actor と critic には，Gaussian soft-max network を用いた．

$$y = \sum_{k=1}^{K} w_k \mathbf{b}_k(\mathbf{x}),$$

$$\mathbf{b}_k(\mathbf{x}) = \frac{\prod_{i=1}^{n} \exp[-(\frac{x_i - c_{ki}}{s_{ki}})^2]}{\sum_{l=1}^{K} \prod_{i=1}^{n} \exp[-(\frac{x_i - c_{li}}{s_{li}})^2]},$$

ここで，$(c_{k1}, \ldots, c_{kn})$ と $(s_{k1}, \ldots, s_{kn})$ は，k 番めの基底関数の中心と大きさをしめす．一般的には，両方

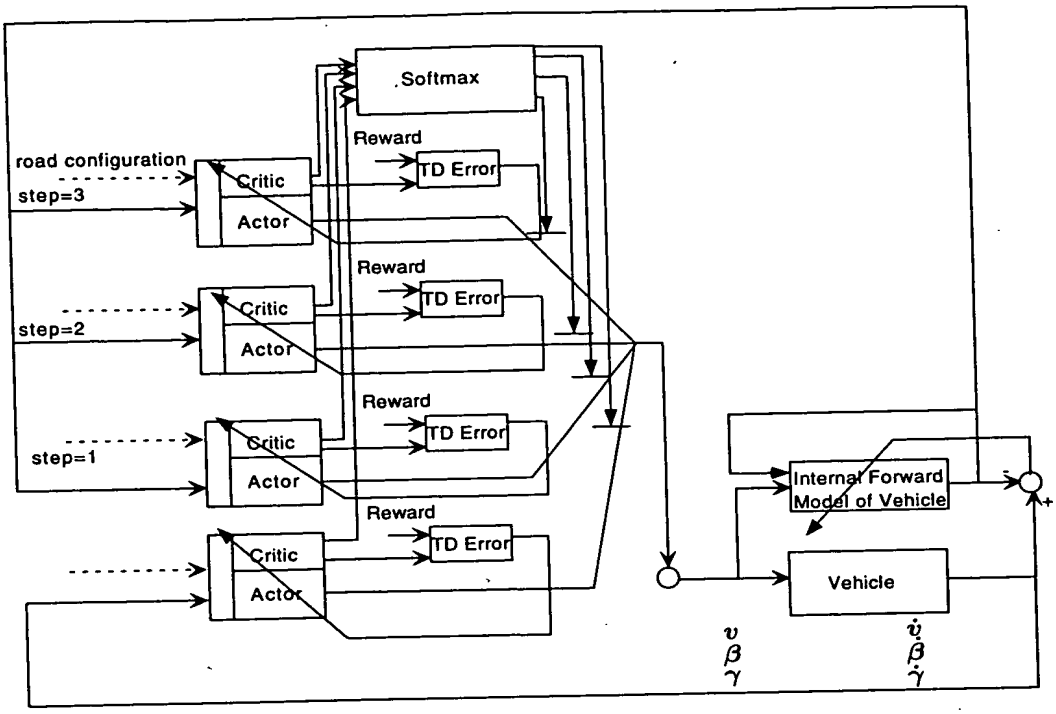

図 1: 内部モデルの予測による Actor-Critic 構造

を調整することが可能であるが，本報告では，計算を簡略化するために格子上に配列することで，両方の値を固定し，w_k だけを調節した．

3.2 自動車のモデル

2 輪車モデルを用いて，自動車の挙動を計算した [13]．

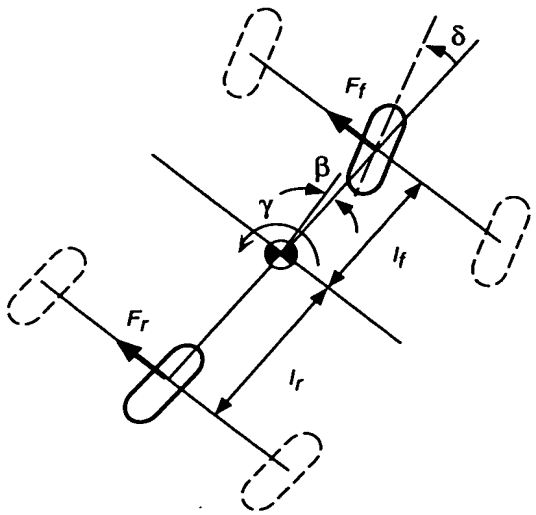

図 2: 4 輪車と等価的な 2 輪車モデル

図 2 に制御対象のモデルを示す．図 2 の前輪操舵

車両の運動方程式は，次式で与えられる．

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$$

$$\boldsymbol{x} = \begin{bmatrix} \beta \\ \gamma \end{bmatrix},$$

$$\boldsymbol{u} = \begin{bmatrix} \delta \\ 0 \end{bmatrix},$$

$$\boldsymbol{A} = \begin{bmatrix} -\frac{c_f + c_r}{MV} & -1 - \frac{l_f c_f - l_r c_r}{MV^2} \\ -\frac{l_f c_f - l_r c_r}{I_z} & -\frac{l_f^2 c_f - l_r^2 c_r}{I_z V} \end{bmatrix},$$

$$\boldsymbol{B} = \begin{bmatrix} \frac{c_f}{MV} & \frac{c_r}{MV} \\ \frac{l_f c_f}{I_z} & -\frac{l_r c_r}{I_z} \end{bmatrix}.$$

ここで，β，γ は，それぞれ，車体のスリップ角とヨーレイトを，M，V，I_z は，それぞれ車両重量，車速，車両ヨーイング慣性モーメントを，l_f（l_r），c_f（c_r）は，前（後）輪–重心間距離，前（後）輪コーナリングパワーを，δ は，前輪実舵角をあらわす．

車両の重心点の座標を (X,Y) とするとその軌跡は，

$$X = X_0 + V\int_0^t \cos(\beta+\theta)dt \tag{14}$$

$$Y = Y_0 + V\int_0^t \sin(\beta+\theta)dt \tag{15}$$

$$\theta = \theta_0 + \int_0^t \gamma dt \tag{16}$$

$$\tag{17}$$

であたえられる．ここで，X_0，Y_0，θ_0 は，それぞれ $t=0$ での値である．

3.3 自動車の運転操作

自動車を運転する時は，前方を注視しながら道路に沿って走行する．したがって，目標軌道としては，道路形状から推定される情報を用いていると考えられる．本報告では，現在の制御入力から将来の位置を動力学モデルを用いて推定し，その場所での誤差を計測する．この誤差を基にあらかじめ決めたルールによって報酬を計算する．

3.4 報酬

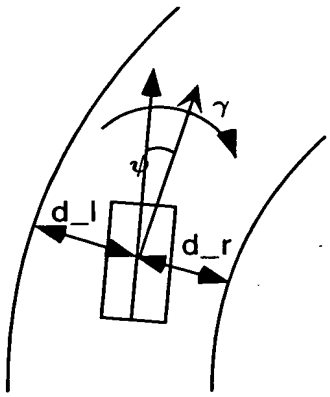

図 3: 道路形状と報酬

図 3 に，曲率一定のカーブを走行中の報酬の情報について示す．ψ は，進行方向と道路のなす角であり，d_l, d_r は，それぞれ，道路の中央からの左右の道路端までの距離である．道路の左右の端からの距離が分かるとその合計で道路の幅が分かることになる．これにより，道路の幅の広いときと細いときで，報酬や制法則が変わる可能性が生じる．一方，報酬は，図 4 により計算する．左右の道路の端までの距離が $1.5m$ 以上であれば報酬は 0，距離が $1.0m \sim 1.5m$ であれば報酬は徐々に減少し，距離が $1.0m$ 以下になると -1 となる．

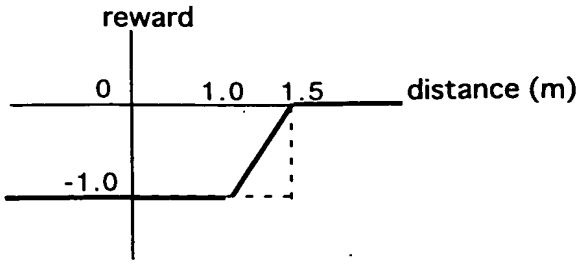

図 4: 報酬の計算

3.5 学習

図 5 に学習に使用したコースを示す．直線とクロソイド曲線と一定曲率のカーブの組合せで，2 種類の一定曲率のカーブを用いて，道路を作成した．

3.6 内部モデルの学習

現在の制御入力から，将来の自動車の状態を推定するためには，動力学モデルが必要であるが，人間は，運転をしながら自然に学習を行なう．本モデルにおいても，現在の制御入力と状態の変化から，3 層の神経回路モデルを用いて学習を行なった．入力は，自動車の速度 v，スリップ角 β，回転角速度 γ，出力は，自動車の加速度 $\dot{v}$，スリップ角の変化 $\dot{\beta}$，回転角速度の変化 $\dot{\gamma}$ である．中間層は 10 個用いた．自動車の制御は，0.05 秒間隔で行なっているが，内部モデルは，0.5 秒後の状態を推定するように学習を行なっている．このとき，勾配項のみを持つ誤差逆伝搬 [14] を用いた場合，評価関数曲面の谷において，重みが振動するため収束が遅くなる．そのため，この問題点を解決した kick out 法 [15] を用いた．

3.7 Actor-Critic の学習

3.4 章で示した報酬をもとに，Actor-Critic の学習を行なった．複数の Actor や Critic は，分配信号をもとに TD 誤差を分配することにより学習を行なった．さらに，各 Actor の出力は，分配信号と掛け合わして，すべての総和を取ることで，全体の制御出力を決定した．

3.8 結果

学習後，未学習の曲率を含むコースを，学習していない速度 ($18[m/sec]$) で走行した軌跡を図 6 に示す．

また，テスト道路を走行中の分配信号の遷移を図 7 に示す．道路上で円をつけた印のところは，道路を $1m$

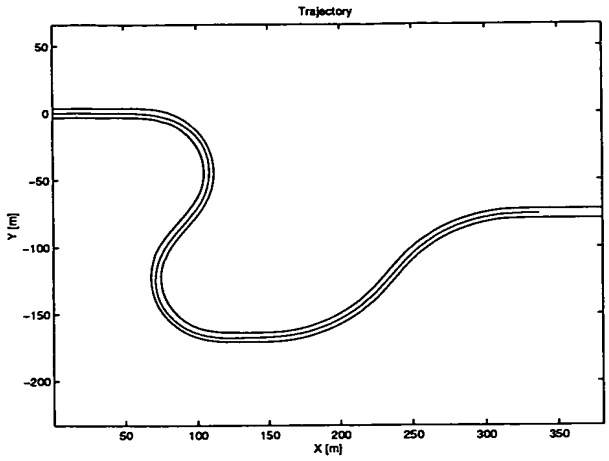

図 5: 学習した道路の走行結果

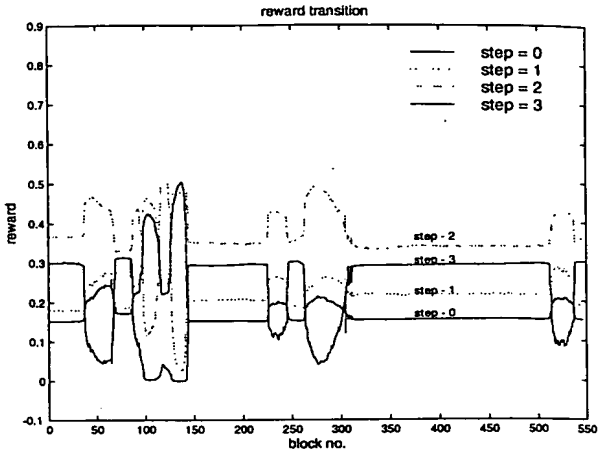

図 7: 報酬の比率の遷移

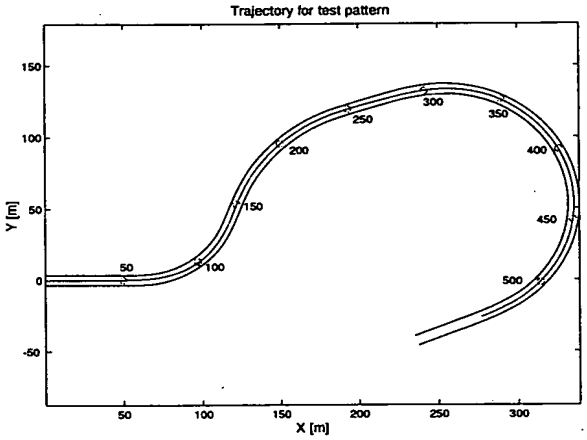

図 6: 未学習な道路走行結果

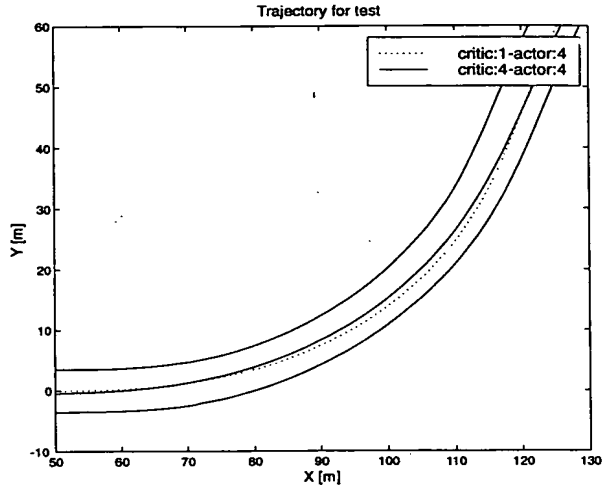

図 8: 未学習な道路走行結果 (異なるアーキテクチャ)

で区切ったブロックの数を示してあり，図 7 の横軸の値となっている．道路幅が狭いときは特に，$step = 3$ の情報は $step = 2$ の情報よりも道路からはずれる場合が多く，評価関数の値が小さくなっていると考えられる．カーブの曲率が小さいところでは，2 秒先でも道路からはみ出してしまうため，分配信号が $step = 0$, $step = 1$ のところで大きくなっている．それ以外は，$step = 2$ が大きい値を示している．

図 8 に Actor-Critic の数の違いによる走行軌跡の違いを示す．critic がひとつだけのモデルでは，道路上中央に自動車がいる時に高い報酬を示すようになるため，遠くの地点で道路から外れる状態では，報酬は小さくなり，遠くの地点での actor の出力はあまりステアリング操作に影響を及ぼさない．このため，全体に道路の中央を走行する軌跡を通るようになる．一方，critic も 4 つ用いたモデルでは，遠くの地点で道路から外れる状態であっても報酬が大きい値になっているため，遠くの地点での actor の出力も考慮され早めにステアリング操作が開始される．このため，より内側を通るような軌跡をとる．

更に，critic も actor も 4 つのモデルを用いて，障害物の回避を行なった結果を図 9 に示す．

図中黒い部分が障害物である．障害物の手前から車の向きを変え始めており，障害物を避け走行できていることがわかる．障害物がある地点では，道幅が減少していることと同じになるため，actor と critic の入力がかわり，近くは，道幅が広い時の actor と critic の学習結果が反映され，遠くでは，道幅が狭い時の actor と critic の学習結果が反映される．このため，遠くで障害物がある状態では critic は，より道路から自動車がそれたとしても高い値を出力しており，その状態の actor の出力が最終的なステアリング操作に影響を与えることになる．

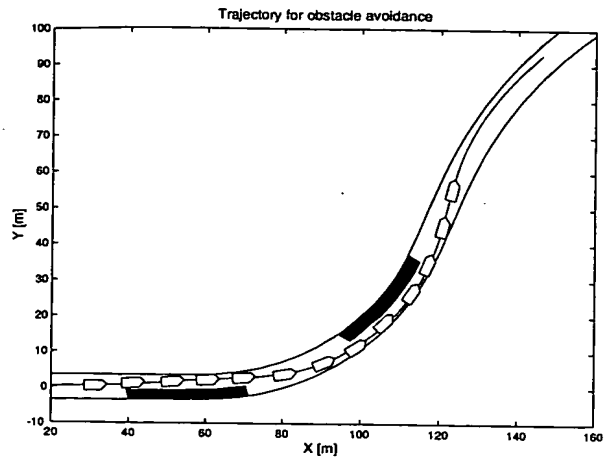

図 9: 障害物回避

4 終りに

本報告では，実時間の TD 学習を用いて，車両のステアリング操作を学習するモデルを提案した．

このモデルの中で，n 秒後の位置を計算するために，車のダイナミクスを直接用いて計算している．しかし，人間がはじめから自動車の順ダイナミクスを獲得しているとは思えないために，何らかの方法で，順ダイナミクスモデルを学習する必要がある．今回は,3 層の神経回路モデルを用いて現在の状態から 0.5 秒後の状態変化を推定するダイナミクスモデルを学習により獲得している．そのため，未学習のデータに対しては正しい状態推定ができる保証はないが，神経回路モデルの汎化能力により，ほぼ正しい出力が得られていた．

さらに，複数の場所での情報から，学習によりどの位置をみて（どこの位置の情報を用いて）ステアリング操作を行えば良いのかを自動的に獲得できた．この視線の変化の様子と実際の運転中に人間が行っている視線の変化を対応させ，より人間の動作に近いモデルが獲得されるように発展させたい．

今回のモデルでは，状態予測を行なう時に制御信号は固定して行なっていた．さらに，離散的に状態予測を行なっていた．人間が実際に状態予測を用いて制御を行なっている場合，どのようになっているのか，複数の状態予測による"actor-critic" をどのように統合しているのかなどは今後の課題である．

参考文献

[1] D.M. Wolpert, R. C. Miall, and M. Kawato. Internal models in the cerebellum. *Trends in Cognitive Sciences*, pp. 338–347, 1998.

[2] M. Haruno, D.M. Wolpert, and M. Kawato. Multiple paired forward-inverse models for human motor learning and control. *NIPS98*, 1998.

[3] 片桐憲一, 銅谷賢治, 川人光男. 複数のモデルを用いた強化学習による非線形制御方式. 信学技報 (NC98-46), pp. 25–32, 1998.

[4] 小池康晴, 銅谷賢治. 強化学習による自動車運転技能の獲得. 信学技報 (NC96-169), pp. 107–114, 1997.

[5] Michael F. Land and D. N. Lee. Where we look when we steer. *Nature*, pp. 742–744, 1994.

[6] Michael F. Land and Julia Horwood. Which parts of the road guide steering? *Nature*, pp. 339–340, 1995.

[7] 三浦利章. 応用実験心理学の立場より-視覚的注意特性と自動車運転 の安全性-. 第 62 回日本心理学会, p. S21, 1998.

[8] R. P. N. Rao, G. J. Zelinsky, M. M. Hayhoe, and D. H. Ballard. Eye movements in visual cognition: A computational study. Technical report, University of Rochester, 1997.

[9] D.M. Wolpert, Z. Ghahramani, and M.I. Jordan. An internal model for sensorimotor integration. *Science*, Vol. 269, pp. 1880–1882, 1995.

[10] R.C. Miall and D.M. Wolpert. Forward models for physiological motor control. *Neural Networks*, Vol. 9, No. 8, pp. 1265–1279, 1996.

[11] Kenji Doya. Temporal difference learning in continuous time and space. In J. M. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 8*. MIT Press, Cambridge, MA, 1996.

[12] R.S Sutton and Barto A.G. *Reinforcement Learning*. MIT Press, 1998.

[13] 安部正人. 自動車の運動と制御. 山海堂, 1992.

[14] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, Vol. 323, pp. 533–536, 1986.

[15] K. Ochiai and S. Usui. Kick-out learning algorithm to reduce the oscillation of weights. *Neural Networks*, Vol. 7, No. 5, pp. 797–807, 1994.